huangzhichong / **selenium-webdriver-cheatsheet.md**

Created 6 years ago

Selenium Webdriver CheatSheet

`selenium-webdriver-cheatsheet.md`

## API workthough

1. Open a browser

```
# start an instance of firefox with selenium-webdriver
driver = Selenium::WebDriver.for :firefox
# :chrome -> chrome
# :ie     -> iexplore
```

- Go to a specified URL

```
driver.get 'http://google.com'
driver.navigate.to 'http://google.com'
```

`NOTE` -- the WebDriver may not wait for the page to load, you'd better using explicit and implicit waits.

- Locating Elements
  - `find_element` -- Find the first element matching the given arguments.
  - `find_elements` -- Find all elements matching the given arguments
  - By ID

```
# example html
# <input id="q">...</input>

element = driver.find_element(:id, "q")
```

  - By Class Name

```
# example html
# <div class="highlight-java" style="display: none; ">...</div>

element = driver.find_element(:class, 'highlight-java')
# or
element = driver.find_element(:class_name, 'highlight-java')
```

  - By Tag Name

```
# example html
# <div class="highlight-java" style="display: none; ">...</div>

element = driver.find_element(:tag_name, 'div')
```

  - By Name

```
# example html
# <input id="q" name='search' type='text'>…</input>

element = driver.find_element(:name, 'search')
```

- By Link Text

```
# example html
# <a href="http://www.google.com/search?q=cheese">cheese</a>

element = driver.find_element(:link, 'cheese')
# or
element = driver.find_element(:link_text, 'cheese')
```

- By Partial Link Text

```
# example html
# <a href="http://www.google.com/search?q=cheese">search for cheese</a>
element = driver.find_element(:partial_link_text, 'cheese')
```

- By XPath

```
# example html
# <ul class="dropdown-menu">
#   <li><a href="/login/form">Login</a></li>
#   <li><a href="/logout">Logout</a></li>
# </ul>

element = driver.find_element(:xpath, '//a[@href='/logout']')
```

  - `NOTE` -- When using Element#find_element with `:xpath`, be aware that,
    - webdriver follows standard conventions: a search prefixed with "//" will search the entire document, not just the children of this current node.
    - Use ".//" to limit your search to the children of the receiving Element.

- By CSS Selector

```
# example html
# <div id="food">
#   <span class="dairy">milk</span>
#   <span class="dairy aged">cheese</span>
# </div>

element = driver.find_element(:css, #food span.dairy)
```

- Element's operation
  - Button/Link/Image

```
driver.find_element(:id, 'BUTTON_ID).click
```

  - Text Filed

```
# input some text
driver.find_element(:id, 'TextArea').send_keys 'InputText'
# send keyboard actions, press `ctral+a` & `backspace`
```

```
driver.find_element(:id, 'TextArea').send_keys [:contol, 'a'], :backspace
```

- Checkbox/Radio

```
# check if it is selected
driver.find_element(:id, 'CheckBox').selected?
# select the element
driver.find_element(:id, 'CheckBox').click
# deselect the element
driver.find_element(:id, 'CheckBox').clear
```

- Select

```
# get the select element
select = driver.find_element(:tag_name, "select")
# get all the options for this element
all_options = select.find_elements(:tag_name, "option")
# select the options
all_options.each do |option|
 puts "Value is: " + option.attribute("value")
 option.click
end

# anthoer way is using the Select class after seleniun-webdriver 2.14
element= driver.find_element(:tag_name,"select")
select=Selenium::WebDriver::Support::Select.new(element)
select.deselect_all()
select.select_by(:text, "Edam")
```

- visibility

```
driver.find_element(:id,'Element').displayed?
```

- get text

```
driver.find_element(:id,'Element').text
```

- get attribue

```
driver.find_element(:id, 'Element').attribute('class')
```

- Driver's operation
  - execute javascript

```
driver.execute_script("return window.location.pathname")
```

  - wait for a specific element to show up

```
# set the timeout to 10 seconds
wait = Selenium::WebDriver::Wait.new(:timeout => 10)
# wait 10 seconds until the element appear
wait.until { driver.find_element(:id => "foo") }
```

  - implicit waits

An implicit wait is to tell WebDriver to poll the DOM for a certain amount of time when trying to find an element or elements if they are not immediately available

```
driver = Selenium::WebDriver.for :firefox
# set the timeout for implicit waits as 10 seconds
driver.manage.timeouts.implicit_wait = 10

driver.get "http://somedomain/url_that_delays_loading"
element = driver.find_element(:id => "some-dynamic-element")
```

- switch between frames

```
# switch to a frame
driver.switch_to.frame "some-frame" # name or id
driver.switch_to.frame driver.find_element(:id, 'some-frame') # frame element

# switch back to the main document
driver.switch_to.default_content
```

- swich between windows

```
driver.window_handles.each do |handle|
  driver.switch_to.window handle
end
```

- handle javascript dialog

```
# get the alert
a = driver.switch_to.alert
# operation on the alert
if a.text == 'A value you are looking for'
  a.dismiss
else
  a.accept
end
```

- Cookies
  - Delete cookies

```
# You can delete cookies in 2 ways
# By name
driver.manage.delete_cookie("CookieName")
# Or all of them
driver.manage.delete_all_cookies
```

**tit** commented on Nov 5, 2013

My way for Select.

```
def select options
    # :class, :class_name, :id, :link_text, :link, :partial_link_text, :name, :tag_name, :xpath
    # :xpath by default
    how = options[:how] || :xpath
```

```
    # example '/html/body/select'
    what = options[:what]

    # random select if nil
    value = options[:value]

    wait = options[:wait] || Selenium::WebDriver::Wait.new
    driver = options[:driver]

    wait.until do
        select = driver.find_element how, what
        select.find_elements :tag_name, 'option'
    end

    select = driver.find_element how, what
    options = select.find_elements :tag_name, 'option'
    random = options.sample.attribute 'value'
    value = value || random

    options.each do |option|
        option.click if value == random
    end
end
```

**The-canary48** commented on Oct 14, 2015

Hi,
I just want to put it simple

## Requirements

select the desired text in a paragraph using selenium

**rameshdh2009** commented on Oct 29, 2015

Create update
Selenium web driver using Java, How to click on create and update. Please help me .

**michael47265** commented on Nov 15, 2015

Great cheatsheet! Just a couple of minor issues I came across:
driver.find_element(:id, 'BUTTON_ID).click #' missing after _ID

driver.find_element(:id, 'TextArea').send_keys [:contol, 'a'], :backspace #contol should be control

**masterkrang** commented on Oct 8, 2017

just wanted you to know i think you're cool

**FernandoZnga** commented on May 29

Hello, Do you know how to count the elements on a ul element? from a table I can use len but won't work on ul element

**BrockTanium** commented on Aug 23

so great this is amaze. thanks much